Docket JP920000411US1

<div align="right">
Appl. No.: 09/783,250<br>
Filed: February 14, 2001
</div>

## REMARKS

Claims 1-6, 9-15, 17-23, 25-36 and 38-44 stand rejected under 35 USC 102(e) based on US patent 6,067, 639 (Rodrigues). Claims 7, 8, 24, 37 and 45 stand rejected under 35 USC 103(a) based on Rodrigues in view of US patent 5,860,009 (Uchihira). Claim 16 stands rejected under 35 USC 103 (a) based on Rodrigues et al. in view of US patent 6,397,378 B1 (Grey). Method claims 1, 2, 3 and 5, computer program product claims 18, 19, 20 and 22, and system claims 31, 32, 33 and 35 have responsively been amended to overcome the rejections. Claims 23 and 36 have also been amended to conform them to their amended base claims. Claims 9, 25 and 38 have been amended to correct informalities.

Claims 1, 18 and 31. Specifically, claim 1, for example, now points out that a method of testing a program includes step a), according to which the program is divided into groups *such that every statement in the program belongs to at least one of the groups*. (The discussion herein of claim 1 also applies to claims 18 and 31, although claims 18 and 31 are for different forms of the invention.) This is herein added to the claims because it is an aspect of how there is a determination based on executed groups that *all* statements have been executed, and because this aspect is not taught by the cited art. Support for this amendment is found in the specification and figures. See page 18 and FIG's 4A - 4C (showing all statements in the program of FIG. 4A, i.e., lines 3, 5, 6, 8 - 18, 20 - 25, 27 and 28, divided into respective groups B11 - B18 in FIG's 4B and 4C, wherein group B11 in FIG. 4B includes lines corresponding to lines 3, 5 and 6 of FIG. 4A, group B12 includes lines corresponding to lines 8 and 9 of FIG. 4A, group B13 in FIG. 4B includes a line corresponding to line 10 of FIG. 4A, group B14 in FIG's 4B and 4C includes lines corresponding to lines 11 - 13 of FIG. 4A, group B15 in FIG. 4C includes lines corresponding to lines 14 - 15 of FIG. 4A, group B16 in FIG. 4C includes lines corresponding to lines 16 - 18 of FIG. 4A, group B17 in FIG. 4C includes lines corresponding to lines 20 - 21 of FIG. 4A, and group B18 in FIG. 4C includes lines corresponding to lines 22 - 25, 27 and 28 of FIG. 4A).

The amended claim 1 goes on to state that *each of the groups contains a respective sequence of ones of the statements* such that all the statements of *such a group* are executed if at least one statement of said group is executed. This amendment makes it clear that it is not the plurality of groups that has the sequence of statements. Rather, each individual group has its own sequence of statements. (It should be understood that the "sequence of . . . statements" is

Docket JP920000411US1                                          Appl. No.: 09/783,250
                                                               Filed: February 14, 2001

not necessarily the same for each group. That is, each group may have its own respective
sequence of statements.) Support for this amendment is found in the specification and figures.
See above reference to page 18 and FIG's 4A - 4C (showing sequences such as the statement of
line 10 of FIG. 4A for group B13 (a sequence of one statement) and the statements of lines 11 -
13 of FIG. 4A for group B14, etc.). The amended claim 1 goes on to state that such a group is
deemed to be executed if at least one of the statements of the group is executed when the
program is executed. For support, refer again to page 18 and FIG's 4A - 4C (showing groups of
statements in which none of the groups have any branching statements, except at the end of the
group, that would cause some of the statements in the group to not be executed along with the
others in the group). With this change, a clearer meaning is provided regarding the executed
group referred to in the next step.

The amended claim 1 goes on to set out step b), according to which there is a determining
of *the ones of the groups* that are executed when said program is executed while testing said
program. See "Overview and Discussion of the Invention," beginning on page 8, "Method to
Ensure Coverage in Testing," beginning on page 13, and FIG. 2. With this amended language,
step b) is more explicitly tied back to the earlier step a) in such a way that now step b) is clearly
not anticipated by art that merely discloses a determination about just any kind of "groups" that
are executed. That is, a "group" as now mentioned in step b) is more clearly the particular kind of
group defined in the preceding step a).

The amended claim 1 goes on to set out step c). In this step unexecuted ones of the
groups are indicated based on the ones of the groups that were determined *in step b)* to have been
executed. This more clearly points out the particular invention than did the former language,
which stated that "unexecuted groups are determined based on said determining." Now it is clear
that the unexecuted groups are determined based on the earlier step, which determined the groups
that have been executed.

Finally, amended claim 1 now states that in step d) a tester is enabled to execute said
unexecuted groups such that said tester can ensure that all statements in said program are
executed at least once. This step has merely been amended to eliminate unnecessary usage of
"plurality of" with reference to the unexecuted groups.

Docket JP920000411US1                                      Appl. No.: 09/783,250
                                                           Filed: February 14, 2001

Applicant contends that the amendments to claims 1, 18 and 31 overcome the rejections. Neither the cited teachings of Rodrigues nor any of the other references disclose or even suggest that a program being tested is divided into groups *such that every statement in the program belongs to at least one of the groups*, and that each of the groups contains a respective sequence of ones of the statements such that *all the statements of such a group are executed if at least one statement of said group is executed*, as stated in amended claim 1. To the contrary, Rodrigues teaches randomly selecting test operations, logging the randomly generated sequence of operations in a playback file. See, for example, column 4, lines 52 - 67; column 9, lines 18 - 49. Rodrigues also teaches how to enable playing back such a randomly generated sequence of test operations. See, for example, column 5, lines 1 - 6. The Office action cites column 15, lines 54 - 65, of Rodrigues. However, the cited portion of Rodrigues concerns defining groups of *test objects*, not groups of the statements of the program under test, as now more clearly defined in the amended claims of the present invention. Moreover, the cited portion of Rodrigues concerns weighting test object groups to influence how they are *randomly* selected. Rodrigues, column 16, lines 1 -15. Applicant contends that this is not relevant to dividing a program being tested into groups of statements such that every statement in the program belongs to at least one of the groups, and such that all the statements of such a group are executed if at least one statement of said group is executed, as per the amended claims.

Also, the cited teachings do not address the issue of ensuring that *all statements* in the program are executed at least once. Accordingly, they do not teach or suggest keeping track of executed groups (for the particular type of claimed group discussed immediately above), deducing from the executed groups which ones of the groups have not been executed, and ensuring that the unexecuted groups get executed, as per amended claim 1. The Office action cites column 13, lines 28 - 35, of Rodrigues. However, the cited portion of Rodrigues concerns determining which elements in a playback file have not been played back. This does not concern determining which groups of the type particularly set out in claim 1 are executed and about indicating unexecuted ones of these groups based on the ones of the groups that were determined to have been executed. Even aside from the particular type of groups, the cited portion of Rodrigues does not even concern *groups*, but rather concerns merely making sure that all "entries" or "test operation objects" that were randomly selected are played back.

Docket JP920000411US1

Claims 2, 19 and 32. Amended claim 2 now states that the method also involves including an extra statement in each of the (previously mentioned) groups, and that the execution of such an extra statement enables said determining in step b) to identify an executed one of the groups corresponding to the extra statement. (The discussion herein of claim 2 also applies to claims 19 and 32, although claims 19 and 32 are for different forms of the invention.) As in amended claim 1, this amended language in claim 2 improves clarity about the "determining" to which the claim refers. Also, while it is true that executing the extra statement enables identifying execution of the extra statement, the amended language is more to the point. That is, identifying that the extra statement is executed enables identifying an executed *group* (and, impliedly, all the group's associated statements). See page 18 and FIG's 4A - 4C ("Instrumentation module 310 may insert program statements . . . While only one program statement is shown inserted for each group, more than one statement also can be inserted as a designer wishes. The effect of the inserted program statement is to pass the group identifier in the statement to coverage tracking module 350 when the statement is executed.").

The Office action cites column 15, lines 60 - 62, of Rodrigues. See the Applicant's response above.

Claims 3, 20 and 33. Amended claim 3 now states that the extra statements contain respective group identifiers, and that the determining in step b) includes examining such a group identifier to determine a specific one of the groups which has been executed. (The discussion herein of claim 3 also applies to claims 20 and 33, although claims 20 and 33 are for different forms of the invention.) Again, the amended language improves clarity about the particular "determining" to which the claim refers. Also, the language makes it clear that each group has its own respective group identifier, and that the "groups" referred to are the *same particular kind of groups* defined in the earlier claims. See page 18 and FIG's 4A - 4C ("Instrumentation module 310 may insert program statements, with each statement containing class name and group identifier as parameters into program code as depicted, for example, in lines 3, 9, 14, 18 of Figure 4B.").

The Office action cites column 15, lines 60 - 62, and column 16, lines 1 - 15, of Rodrigues. See the Applicant's response above.

Claims 5, 22 and 35. Amended claim 5 now states that the method also involves grouping a *sequence of the groups* into a block. (The discussion herein of claim 5 also applies to claims 22 and 35, although claims 22 and 35 are for different forms of the invention.) Also, it states that the method includes *determining* that the block has been executed only if all of the groups of the block are executed. This amended language makes it clear once again that the "groups" referred to are not just any kind of groups, but rather are the *same particular kind of groups* defined in the earlier claims. Aside from making it clear that there is an antecedent to "groups," the amended language about "grouping a sequence of the groups into a block" is not intended to convey a different meaning than "grouping a plurality of sequential groups into a block," but is merely intended to be a more direct and clear statement. Also, the "indicating" in claim 1 was about *unexecuted* groups, and this claim refers to a number of *executed* groups, i.e., an executed block. In claim 1, there is a step involving *determining* the ones of the groups that are *executed*. For consistency, the term "indicating" in claim 5 has therefore been changed to "determining."

The Office action cites column 9, lines 62 - 65. This portion of Rodrigues concerns the *test object* groups referred to in the previously cited portions. See the Applicant's response above.

Claims 4, 6- 17, 21, 23 - 30, 34 and 36 - 45. These claims all depend on claims that Applicant contends are patentably distinct, as discussed herein above. This basis alone is sufficient such that these claims 4, 6- 17, 21, 23 - 30, 34 and 36 - 45 are patentably distinct. Moreover, Applicant contends that these claims particularly set out patentable features that the cited art does not actually teach or suggest. In addition, the rejection of claims 7, 8, 24, 37 and 45 under 35 USC 103 rely upon the combination of Rodrigues and Uchihira. However, the Office action has provided no basis for the suggestion or motivation to combine the references, as required for a rejection of this kind. MPEP 2143.01 ("Suggestion or Motivation To Modify the References"). Further, Uchihira concerns the support of parallelization by means of serializing concurrent programs. The Office action apparently equates *processes* that are serialized according to the teaching of Uchihira to the *blocks* of the present invention. However, the blocks according to the present invention are of a type that at a higher level potentially *contain* one or more blocks at a lower level. Page 19 (discussing step 520). Applicant contends

Docket JP920000411US1                                  · · Appl. No.: 09/783,250
                                                          Filed:  February 14, 2001

that serializing processes so that one process executes before a next process is contrary to

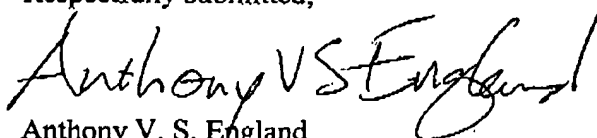forming blocks such that one block contains another.

## PRIOR ART OF RECORD

Applicant has reviewed the prior art of record cited by but not relied upon by Examiner,

and asserts that the invention is patentably distinct.

## REQUESTED ACTION

Applicant contends that all the claims of the invention in accordance with amendments

submitted herein are patentably distinct, and hereby requests that Examiner grant allowance and

prompt passage of the application to issuance.


Respectfully submitted,

Anthony V. S. England
Attorney for Applicants
Registration No. 35,129
512-477-7165
a@aengland.com